David Cain

Notes:

* When putting together I found that due to the ESP8266 operating off 3.3V led to the LEDs not being able to interpret the commands from the controller on the receiver end.
  + The solution to this was to have a ‘Buffer LED’ to interpret the commands on a lower voltage, then this LED would send proper signals to the remainder of the strip
* Since I have used these LEDs before, I was hoping to just carry over an animation library that would act as an idle display for the LEDs. Unfortunately, the library I was intending to use, does not support these controllers. This led to me having to hard code 5 animations that take more time than I was hoping for. I plan to have roughly 5-10 animations for day of presentation.

Demonstration Plans:

My goal when presenting this project will of course to display that one controller will be interpreting audio signals and acting as a Wi-Fi access point. There will be at least 2 other controllers that will be connected to this Wi-Fi access point and controlling their own LED strips. Since there is a decently working prototype, the goal is to come up with a decent program that will react to the excitement of the music and shift towards a certain color spectrum. I.E. if a music seems to have a less lively beat, shift towards the blue spectrum or if there is more excitement, shift to a red centered spectrum. Again, the goal is to have 5-10 operation modes, or animations, but what is up in the air is how to interact with the main hub to allow the user to switch between the modes. So far there is a simple button system that works decently based off long presses.

Considering the hub needs to be portable enough for it to be moved from sound source to another, these is a need for a simple battery management system. Luckily there are plenty of boards and ICs take care of this only leaving you to just properly wire them together. The receivers will be powered off of a DC power supply, meaning these will be mostly stationary.

I have been looking into the possibility of accessing the hub controller via a static IP. The goal of this is that if there are no physical buttons to implement then the microphone portion can be shoved in the back and forget while still having access to control the different operation modes. I’m not very familiar with how this process is going to work out so I’m not necessarily planning on it being there, I just thought it was an interesting idea and will look into the plausibility.

Schematics: